Chapter 7
Object-Oriented Architecture for Web Services Eventing

Krzysztof Ostrowski
Cornell University, USA

Ken Birman
Cornell University, USA

Danny Dolev
The Hebrew University of Jerusalem, Israel

ABSTRACT
Existing web service notification and eventing standards are useful in many applications, but they have serious limitations that make them ill-suited for large-scale deployments, or as a middleware or a component-integration technology in today’s data centers. For example, it is not possible to use IP multicast, or for recipients to forward messages to others, scalable notification trees must be setup manually, and no end-to-end security, reliability, or QoS guarantees can be provided. This chapter proposes an architecture based on object-oriented design principles that is free of such limitations, extremely modular and extensible, and that can serve as a basis for extending and complementing the existing standards. The new approach emerges from the authors’ work on Live Distributed Objects, a new programming model that brings object-orientation into the realm of distributed computing.

INTRODUCTION
Motivation
Notification is a valuable, widely used primitive for designing distributed systems. The growing popularity of RSS feeds and similar technologies shows that this is also true at the Internet scales. The WS-Notification (Graham et al., 2004) and WS-Eventing (Box et al., 2004) standards have been offered as a basis for interoperation of heterogeneous systems deployed across the Internet. Unlike RSS, they are subscription-based and hence free of the scalability problems of polling, and they support proxy nodes that could be used to build scalable notification trees. Nonetheless, they embody restrictions that make them unsuitable as a middleware technology in large-scale systems:
Object-Oriented Architecture for Web Services Eventing

- **No forwarding among recipients.** Many content distribution schemes build overlays within which content recipients participate in message delivery. In current web services notification standards, however, recipients are passive, limited to data reception. Given the tremendous success of peer-to-peer technologies such as BitTorrent for multicast file transfer, one could imagine a future event notification system that uses a BitTorrent-like protocol for transferring multimedia content. But BitTorrent depends on direct peer-to-peer interactions by recipients.

- **Not self-organizing.** While both standards permit the construction of notification trees, such trees must be manually configured, and require the use of proxies, dedicated infrastructure nodes, to build forwarding trees. Our vision would require the ability to automatically setup such trees, either by means of a protocol running directly between the clients or by leveraging external naming and membership services, but the current standards preclude this possibility.

- **Weak reliability.** Reliability in the existing schemes is limited to per-link guarantees resulting from the use of TCP. Many applications, including our live distributed object model, require end-to-end guarantees, and sometimes of strong flavor, e.g. to support virtually synchronous, transactional or state-machine replication. Because receivers are assumed passive and cannot cache, forward messages or participate in multi-party protocols, even very weak end-to-end guarantees cannot be provided.

- **Difficult to manage.** It is hard to create and maintain an Internet-scale dissemination structure that would permit any node to serve as a publisher or as a subscriber, for this requires many parties to maintain a common infrastructure, agree on standards, topology and other factors. Any such large-scale infrastructure should respect local autonomy, whereby the owner of a portion of a network can set up policies for local routing, the availability of IP multicast etc.

- **Inability to use external multicast frameworks.** The standards leave it entirely to the recipients to prepare their communication endpoints for message delivery. This makes it impossible for a group of recipients to dynamically agree upon a shared IP multicast address, or to construct an overlay multicast within a segment of the network. Such techniques are key to achieving high performance and scalability, and could be used to provide QoS guarantees or to leverage emergent technologies, e.g. scalable peer-to-peer overlays for streaming multimedia content.

One could imagine extending WS-Notification or WS-Eventing with the ability to forward data or add explicit support for IP multicast, but ultimately, there is a limit to how much one can achieve with a uniform scheme enforced globally across a heterogeneous system, such as today’s Internet. The path to high performance and scalability leads through a composable, modular architecture, one that respects the diversity and the democratic nature of the Internet and can leverage different, locally optimized protocols for different types of content, or within different parts of the network.

In this article, we propose a principled approach to web service notification in large-scale systems, free of the limitations listed above, modular, and highly extensible. The design presented here is a basis for Live Distributed Objects (Ostrowski, Birman & Dolev, 2008), a novel object-oriented programming model and development environment for large-scale distributed systems. While this architecture is inspired by our work on Live Distributed Objects, it is designed to be generic, and it is compatible, in general, with a wide range of existing protocols, interfaces, and standards.